

LU12d - Datenschutz in Datenbanken

Ziel dieser Seite Ein **praxisnaher Überblick** für Mediamatik-Lernende: Was bedeutet **Datenschutz** konkret bei MySQL? Was ist in der Schweiz (**DSG**) sowie im EU-Kontext (**DSGVO**) zu beachten? Welche **technischen & organisatorischen Massnahmen (TOM)** sind sinnvoll?



Hinweis (keine Rechtsberatung): Diese Unterlage dient der **Sensibilisierung** und **Praxisorientierung** im Unterricht. Für verbindliche Auskünfte bitte interne Richtlinien/Datenschutzbeauftragte/Jurist:innen konsultieren.

1) Warum Datenschutz in der DB wichtig ist

- Datenbanken enthalten oft **Personendaten** (*PII – Personally Identifiable Information*), z. B. Namen, E-Mail, IPs, Bestell- oder Gesundheitsdaten.
- Verstösse gefährden **Betroffene** (Missbrauch, Profiling) und **Organisationen** (Reputation, Bussen, Kosten).
- WordPress & Co.: Häufig **viel PII** (User-Accounts, Kommentare, Formulare) – Standard-Installationen brauchen **Härtung**.

2) Rechtlicher Rahmen (Kurzüberblick)

- **DSG (Schweiz):** Gilt ab 2023 revidiert. Kernprinzipien: **Rechtmässigkeit, Verhältnismässigkeit, Zweckbindung, Datensicherheit, Transparenz**. Rechte der betroffenen Personen (Auskunft, Berichtigung, Löschung, Datenportabilität in gewissen Fällen).
- **DSGVO (EU):** Vergleichbar, teils **strenger** (u. a. Büssen). Gilt, wenn EU-Personen betroffen sind oder EU-Bezug besteht (z. B. Webangebot).
- **Rollen: Verantwortlicher** (Controller) bestimmt Zweck/Mittel; **Auftragsverarbeiter** (Processor) verarbeitet im Auftrag (z. B. Hosting).



Merke: Recht klärt das «**Was & Warum**», Technik (MySQL/Architektur) liefert das «**Wie**» (Sicherheit, Zugriffskontrolle, Logs, Backups).

3) Datenarten & Schutzbedarf

Kategorie	Beispiele	Schutz-Niveau
PII (Personendaten)	Name, E-Mail, IP, Benutzername	Mittel-hoch

Kategorie	Beispiele	Schutz-Niveau
Besonders schützenswerte Daten	Gesundheit, Religion, politische Meinung	Hoch
Geschäftsgeheimnisse	Preise intern, Prototypen	Mittel-hoch
Betriebsdaten	Log- & Metrikdaten (können PII enthalten)	Abhängig vom Inhalt

4) Grundsätze (DSG/DSGVO) in die Praxis übersetzen

- **Datenminimierung:** Nur speichern, was nötig ist (z. B. keine unnötigen Felder).
- **Zweckbindung:** Daten nur für definierte Zwecke verwenden (kein «Sammeln auf Vorrat»).
- **Richtigkeit & Aktualität:** Korrekte, aktuelle Daten (Korrekturen erlauben).
- **Speicherbegrenzung: Lösch-/Archivfristen** umsetzen (auch in Backups!).
- **Integrität & Vertraulichkeit:** Zugriffsschutz, Verschlüsselung, Härtung.
- **Rechenschaftspflicht:** Nachweisbar dokumentieren (wer hat wann was getan).

5) Technische & organisatorische Massnahmen (TOM) für MySQL

5.1 Zugriff & Rollen (Least Privilege)

- **Eigene DB-User** je Anwendung/Team (nicht root).
- **Rollen/Grants** gezielt: **RO** (read-only) fürs Reporting, **RW** (read-write) fürs Backend.
- **Host einschränken** (user '@' localhost statt %).

```
-- Beispiel: minimaler App-User (nur SELECT/INSERT auf produktivem Schema)
CREATE USER 'app_ro'@'localhost' IDENTIFIED BY 'Str0ng!Pass';
GRANT SELECT ON prod_db.* TO 'app_ro'@'localhost';
```

5.2 Starke Passwörter & Policies

- **validate_password**-Komponente aktivieren (Länge, Komplexität).
- **Rotation/Ablauf & Account-Lock** nach Fehlversuchen nutzen.

```
-- Passwortrichtlinie (Beispiel; Anpassung je nach Policy)
INSTALL COMPONENT 'file://component_validate_password';
SET PERSIST validate_password.length = 12;

-- Ablauf/Lock pro Benutzer
```

```
ALTER USER 'app_rw'@'localhost' PASSWORD EXPIRE INTERVAL 180
DAY;
ALTER USER 'app_rw'@'localhost' FAILED_LOGIN_ATTEMPTS 6
PASSWORD_LOCK_TIME 2;
```

5.3 Verschlüsselung - Transport & Ruhe

- **Transport (in transit): TLS/SSL** erzwingen (Client↔Server).
- **Speicher (at rest): InnoDB-Tablespace Encryption** + Keyring.

Transport erzwingen (Server)

```
# my.cnf (Beispiel)
require_secure_transport = ON
ssl_cert = /etc/mysql/ssl/server-cert.pem
ssl_key = /etc/mysql/ssl/server-key.pem
ssl_ca = /etc/mysql/ssl/ca.pem
```

At-Rest (InnoDB)

```
# my.cnf
early-plugin-load = keyring_file.so
keyring_file_data = /var/lib/mysql-keyring/keyring
innodb_encrypt_tables = ON
innodb_encrypt_log = ON
```

```
-- Schlüsselrotation
ALTER INSTANCE ROTATE INNODB MASTER KEY;
```

5.4 Protokollierung & Auditing (sparsam mit PII)

- **Kein** dauerhaftes General Log in Produktion (PII-Risiko, Performance).
- **Audit** gezielt (Wer/Was/Wann), Query-Parameter nicht im Klartext, Retention mit Löschfristen.
- Zugriff auf Logs einschränken.

5.5 Backups & Wiederherstellung

- **Verschlüsselte** Backups (z. B. mysqldump | gpg).
- **Restore regelmässig testen** (auch Rechte/Versionen).
- **Aufbewahrungsfristen + Löschkonzept** (auch in Off-Site/Cloud).
- **Anonymisierte** Backups für Dev/Schulung (keine produktiven PII).

```
# Beispiel: verschlüsseltes Dump
mysqldump --single-transaction --routines prod_db \
| gpg --symmetric --cipher-algo AES256 \
> prod_db-$(date +%F).sql.gpg
```

5.6 Datenmaskierung/Anonymisierung (für Test & Analytics)

- **Views** zur Spaltenmaskierung, **Hashes/Pseudonyme** für E-Mails, **Rauschen** für Metriken.

```
-- Beispiel: Sicht mit maskierter E-Mail
CREATE OR REPLACE VIEW v_users_masked AS
SELECT user_id,
       display_name,
       CONCAT(LEFT(email,2), '***@', SUBSTRING_INDEX(email, '@', -1))
AS email_masked
FROM users;

-- Pseudonymisierung (one-way Hash)
UPDATE users SET email = SHA2(email,256) WHERE /* in
Dev/Copy */ 1=1;
```

5.7 WordPress-spezifische Hinweise

- **DB-User mit Minimalrechten:** Für WP-Betrieb **kein root**, nur benötigte Rechte auf **WP-Schema**.
- **Core/Plugins/Themes** aktuell halten; nur **notwendige** Plugins; regelmässig **Sicherheitsupdates**.
- **wp-config.php** härten: **AUTH_KEYS/SALTS** setzen, **DISALLOW_FILE_EDIT** aktivieren.
- **HTTPS erzwingen** (HSTS), **Adminbereich schützen** (2FA, IP-Restriktion).
- **Formulare/Kommentare:** PII-Felder minimieren; IP-Adressen **anonymisieren**; klare **Privacy Policy**.
- **Backups & Logs:** enthalten oft PII → **verschlüsseln**, Retention definieren, Zugriff beschränken.

6) Prozesse & Organisation

- **Auskunft/Berichtigung/Löschung:** Abläufe & Fristen definieren (auch für DB-Backups & Off-Site-Kopien).
- **Breach-Response:** Erkennen, Melden, Beheben, Dokumentieren (DSG/DSGVO Fristen).
- **Verzeichnis von Verarbeitungstätigkeiten** pflegen (wer verarbeitet was, wo, warum, wie lange?).

- **Auftragsverarbeitung** (Hosting/Cloud): Verträge mit **SCCAVV prüfen, Speicherort/Transfer klären.** ===== **7) Datenschutz-Quickcheck (für Projekte)** ===== - ☐ Minimaldaten? (nur notwendige Felder) - ☐ DB-User getrennt, Least Privilege **umgesetzt** (RO/RW/Rollen)? - ☐ TLS **aktiv & erzwungen** (`require_secure_transport`)? - ☐ At-Rest-Verschlüsselung (**InnoDB + Keyring**)? - ☐ Passwortrichtlinie & Rotation **aktiv** (`validate_password`, `PASSWORD EXPIRE`)? - ☐ Backups verschlüsseln, Restore getestet, Retention **definiert**? - ☐ Logs/Audit: **nur notwendige Daten, Zugriff limitiert, Löschfristen**? - ☐ Anonymisierte Testdaten **statt produktiver PII in Dev/Schulung**? - ☐ WP-Härtung (**Updates, SALTS, kein root, HTTPS, 2FA**)? - ☐ Prozesse für **Auskunft/Löschung/Breach vorhanden & dokumentiert**? ===== **8) Glossar (kurz)** ===== ^ **Begriff** ^ **Erklärung** ^ | PII | ***Personally Identifiable Information*** - **Personendaten** (z. B. Name, E-Mail, IP). | | TOM | **Technische & organisatorische Massnahmen (Sicherheitsmassnahmen)**. | | At Rest / In Transit | **Daten** im Speicher / während der Übertragung. | | Least Privilege | **Nur minimal notwendige Rechte vergeben.** | | AVV/SCC** | Auftragsverarbeitungsvertrag / Standardvertragsklauseln (EU). |

From:

<https://wiki.bzz.ch/> - **BZZ - Modulwiki**

Permanent link:

https://wiki.bzz.ch/modul/m290_guko/learningunits/lu12/theorie/d_datenschutz

Last update: **2025/11/12 00:25**

